Unifying Heterogeneous Distributed Clinical Data in a Relational Database

Keith A. Marrs, Sherry A. Steib, Charlene A. Abrams and Michael G. Kahn
Division of Medical Informatics, Department of Medicine
Campus Box 8005, Washington University School of Medicine, St. Louis MO 63110

Access to clinical data which are distributed among multiple satellite information systems is crucial to delivering better care and reducing costs in many hospitals and medical centers. An integrated view of these data is needed to reduce the effort of users requiring data from multiple systems. We have addressed the issue of distributed data integration while developing both production and research decision-support applications. We describe an ideal integration solution, obstacles to realizing this solution, and our integration requirements and architecture. Our focus is a description of our specific schema and data integration techniques. We conclude with an analysis of our approach.

INTRODUCTION

Due to widely diverse information needs of specialized ancillary services, hospitals frequently have multiple satellite information systems in addition to a centralized main data depository [1]. These clinical data are often distributed among a number of heterogeneous hospital database systems for reasons of security, decentralized control, and application distribution. Access to clinical data which are distributed among these specialized systems is crucial to delivering better care and reducing costs in many hospitals and medical centers [2, 3]. For example, Annevelink describes the need for integration of laboratory, pharmacy, dietary, radiology, and other hospital information systems to support an integrated physician workstation [4]. Our work in hospital quality assurance has required access to microbiology, pharmacy, admission, laboratory, dietary, and other data which exist in various databases and in various data formats. Without an integrated view of clinical data, users must be proficient in several database systems and have an understanding of the data model, language and schema of each system.

We have addressed the issue of integrating clinical data derived from multiple heterogeneous distributed databases while developing both production and research decision-support applications. By moving the database schema integration and system heterogeneity concerns from the application to an integrated database which provides a global, unified view of these distributed data, we have simplified application development and user access. We describe an ideal integration solution, obstacles to realizing this solution, and our integration requirements and architecture. The key focus of this paper is a description of our specific schema and data integration techniques. We conclude with an analysis of our approach.

BARRIERS TO AN IDEAL SOLUTION

A user of distributed data should be unaware of the origin of the information. Logically the user should think she is dealing with a single database without regard for the specifics of the various underlying databases. A heterogeneous, distributed database management system (HDDBMS) provides this virtual database or uniform interface in the form of an integrated view, which hides the structural differences of the underlying databases. The integrated view is usually presented as a global schema, expressed in some common data model such as the Entity-Relationship or Relational model (Figure 1A) [5]. An HDDBMS maps data and operations between the virtual database and the local databases, resolves the differences among the multiple data models, schemas, and systems, and manages distributed transactions and concurrency control.

The federated database model is a variation of the HDDBMS model in which there are multiple global schemas, each of which may be an integration of only a subset of the underlying databases (Figure 1B) [6, 7]. The federated model is more flexible since it supports autonomy of the underlying databases and does not require an all encompassing global schema [5]. For decentralized organizations, this model is ideal because each component database controls the access to its data (specified in its export schemas), providing a decentralized mechanism for data control and security.

0195-4210/92/$5.00 © 1994 AMIA, Inc.

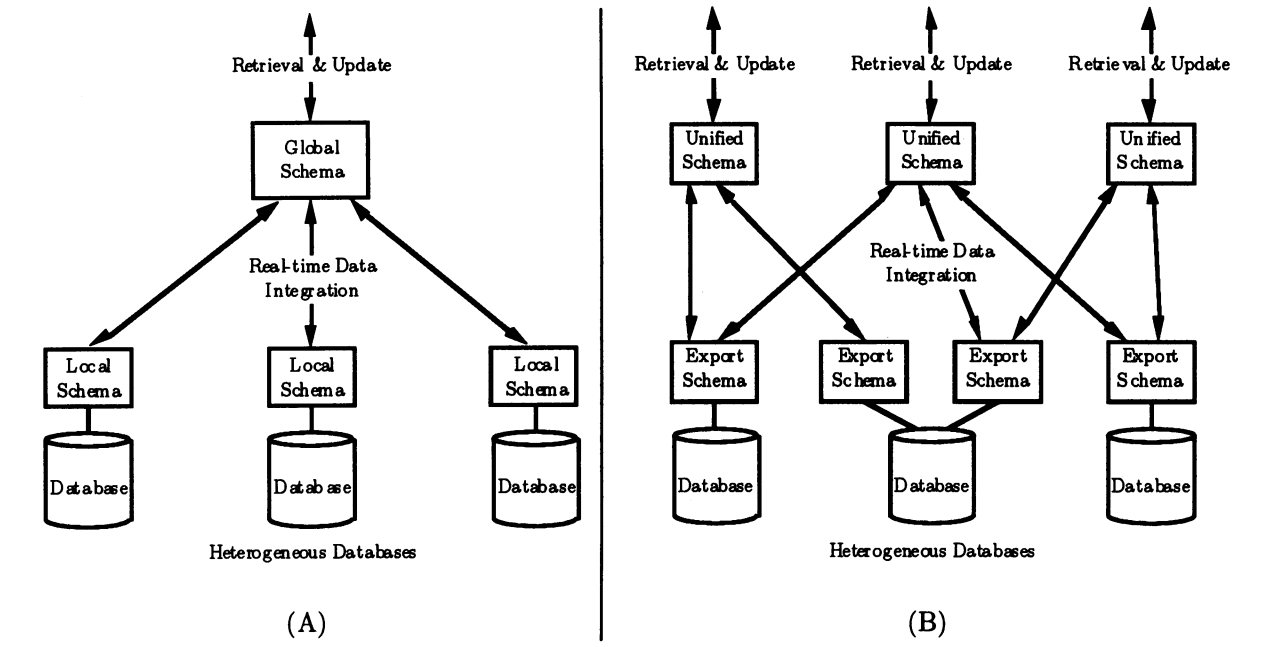

Figure 1: (A) Heterogeneous Distributed Database Model. (B) Federated Database Model.

As evident by the small number of commercial HDDBMSs [8], there are several substantial technical issues to be resolved before HDDBMSs are widely accepted. Some of these issues include optimizing performance across the distributed databases, distributing system load, distributed transaction management and concurrency control, and operation transformation [9]. Even with these difficult unresolved issues, the more difficult barriers to a seamless integration of distributed databases are political, such as administrative control, data security, legacy applications, initial costs, and reluctance to change. Faced with these technical and administrative obstacles, less-than-ideal alternative solutions must be designed for the interim.

INTEGRATION ARCHITECTURE

Although the ideal HDDBMS model discussed above is most desirable, our hospital-based quality assurance applications do not require the real-time, integrated access to data HDDBMS models provide. Our applications do require a single integrated data interface, ad hoc query capabilities, and the persistence and update of processed results. As an example, one application for infection control surveillance requires finalized microbiology culture results [10]. This task does not require real-time data access; the previous day's data is sufficient. Not all clinical decision-support systems which require distributed data could be effective with this constraint. For many systems, real-time data access would be mandatory for timely and accurate recommendations.

The architecture for our solution is shown in Figure 2. We integrate only the necessary data from the underlying databases into a global schema, which is implemented as a set of relations and constraints (i.e., rules, triggers, and stored procedures) in a relational DBMS. Each night, data from the underlying databases are retrieved, parsed, and merged with existing data in the RDBMS. The applications and users have real-time access and update capabilities on this RDBMS, but updates to data are not propagated to the underlying databases, since it is not necessary for our purposes. Updates are made only to processed results and application-specific data instead of underlying primary data.

SCHEMA TRANSLATION AND INTEGRATION

Two basic steps in integrating heterogeneous database systems are schema translation and schema integration. Schema translation involves

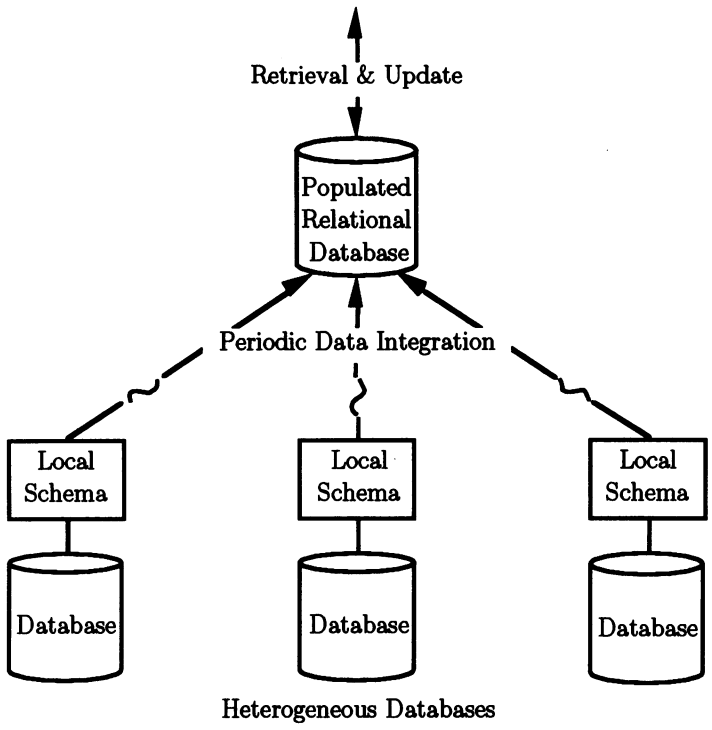

Figure 2: Integration architecture with populated relational database.

mapping schemas between data models, such as from the network model to the relational model. Schema integration is the combining of local database schemas into a single unified, global schema.

Performing schema translation from most data models to the relational model is straight forward except for recursive relationships and transitive closure operations, but the reverse mapping is much more complex. Since we map only into the relational model, this process is relatively straightforward. The underlying databases from which we retrieve data include RDBMSs, virtual sequential access method (VSAM) files, and generated reports. Translation to the relational model is handled during retrieval and parsing of data. We have not addressed the issues of recursive relationships and transitive closure operations because these issues are not present in the clinical data we have integrated.

Batini and Kamel provide good descriptions of the steps in schema integration, which include translating the local schemas into common-model local schemas, defining export schemas for each local schema, identifying and resolving conflicts among these schemas, and merging them into a unified global schema [11, 12]. Schema integration is one of the most difficult aspects of integrating distributed databases because the underlying databases often were developed to satisfy local needs without any consideration for the long-term need to share data; this results in incompatibilities such as conflicting data and definitions, redundant data, and differences in scales/units [3, 5]. Because of these incompatibilities and lack of online data dictionaries, automation of schema integration is probably impossible.

The most laborious step in schema integration is that of discerning the intended or implied semantics of legacy data systems. Yet without a detailed knowledge of the semantics of the data, it is difficult to recognize data elements which represent similar or related concepts. For example, one clinical database contains a field named Patient Account Number whereas a second clinical database contains a field named Patient Registration Number. Do Account Number and Registration Number refer to the same entity? Is there a dependency between the value in Account Number and the value in Registration Number? The age, lack of documentation, and personnel rollover associ-

ated with these systems all contribute to poorly defined schemas.

To capture the semantics of the underlying databases, we employ one of two techniques. In the first technique, we acquire available documentation and sample data; we analyze this information, develop questions, and interview the local data manager and users. With poorly documented systems, this process is iterative, time consuming, and often inaccurate because of special cases missing from documentation and sample data. The second technique relies on printed reports generated from the local database. Because semantics of the database often are embedded within applications and the reports generated by these applications are intended to be "user friendly," it is often simpler to capture semantics by parsing the reports. We used this technique to analyze the data semantics for an expert system which reviews microbiology culture reports to detect hospital-acquired (nosocomial) infections [10]. Instead of integrating the primary data from the laboratory database as described in the first technique, we treat the printed culture reports as the database since they contain all of the laboratory information necessary for the expert system. Understanding the data semantics from these reports proved much easier than analyzing the laboratory database. Although this technique has been effective in our architecture where access to real-time data is not required, capturing semantics from reports obviously is not an option in an ideal HDDBMS or federated system. Also, since reports are subject to change at least as often as the underlying schema, frequent changes could be required to mapping programs.

DATA INTEGRATION

In an ideal HDDBMS, once the global schema and data mappings from the local schemas are defined, the system handles the integration of data for user operations on the global schema. However, in our architecture, data integration routines must be defined for each underlying database and performed on a daily basis. Figure 3 illustrates the data integration steps for the two schema integration techniques discussed above. In a typical scenario, a batch job executes queries each night against a local database. The results of these queries are stored in an ASCII file which later is transferred from the local machine to our machine. If necessary, the file is parsed before being loaded into a temporary table in the RDBMS. Finally, the data in the temporary table is merged into existing tables using SQL scripts. This processing is performed at night to reduce the system load during normal working hours and to avoid affecting interactive users.

Many problems can arise during data integration, including failure of batch jobs on the local systems, network problems during file transfer, and errors during parsing, loading, or merging the data. We have encountered all of these problems and have been able to handle them with little delay of service. The key to our success in handling these problems has been excellent organization and documentation of processes, including daily system backups.

DISCUSSION

Our architecture supports applications which require access to distributed data but do not require either real-time access or update capability. We are using this approach for several deployed and prototyped decision-support applications. In our approach, a global schema is defined as an integration of subsets from the underlying hospital information systems and implemented in a commercial RDBMS; data from the underlying systems are retrieved, manipulated, and merged with existing data in the RDBMS every night; and users and applications retrieve and manipulate integrated data from the integrated RDBMS.

Most of the difficulties in our approach and in any HDDBMS are in understanding the semantics of legacy systems and in defining global schemas. We have shown that it is sometimes simpler to discern the semantics of data by parsing reports generated from the data. We realize that to support many other applications, our architecture will need to evolve into a HDDBMS or federated system. Both the technical and political problems must be solved before this evolution can occur. With ongoing research and the development and acceptance of standards (e.g., RDA, ODBC, IDAPI), solutions to the technical problems appear near [13]. Solving the political problems will take more time and effort but will be easier once systems which support local autonomy and security are available.

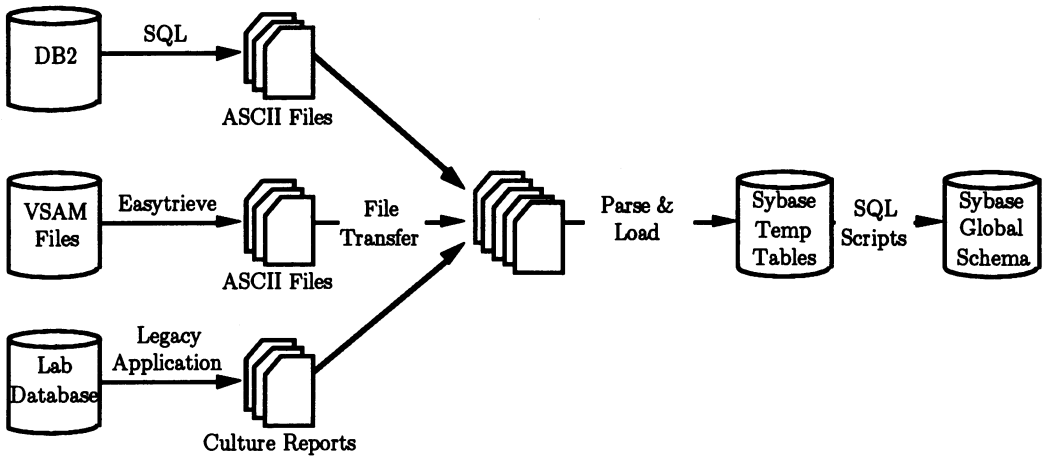

Figure 3: Data Integration Steps.

ACKNOWLEDGEMENTS

Carl Carpenter wrote the first report-parsing prototype. The programming excellence of Bridget Spitznagel is acknowledged. This work is supported by NLM Grant 5-R29-LM05387, Office of Human Genome Research Grant RO1-HG00223, and NCI Contract N01-CM-97564.

Reference

[1] Blum BI. *Clinical Information Systems*. New York NY: Springer-Verlag, 1986.

[2] Institute of Medicine. Committee on Improving the Patient Record. *The Computer-Based Patient Record: An Essential Technology for Health Care*. Washington DC: National Academy Press, 1991.

[3] Haug PJ, Pryor TA, Frederick PR. Integrating radiology and hospital information systems: The advantage of shared data. In: Frisse ME, ed. *Proceedings Symposium on Computer Applications in Medical Care*. New York, NY: IEEE Computer Society Press, 1992:187–91.

[4] Annevelink J, Young CY, Tang PC. Heterogenous database integration in a physician workstation. In: Clayton PD, ed. *Proceedings Symposium on Computer Applications in Medical Care*. New York, NY: McGraw Hill, 1991:368–72.

[5] Sheth AP. Integrating heterogeneous distributed databases: Requirements, concepts and solutions. Course 480: Fall Joint Computer Conference, 1987.

[6] Hsiao DK. Tutorial on federated databases and systems (Part I). The VLDB Journal 1992; *1*: 127–79.

[7] Hsiao DK. Tutorial on federated databases and systems (Part II). The VLDB Journal 1992; *1*: 285–310.

[8] Thomas G, Thompson GR, Chung CW, et al. Heterogeneous distributed database systems for production use. ACM Computing Surveys 1990; *22*: 237–66.

[9] Garcia-Molina H. Research directions for distributed databases. SIGMOD Record 1990; *19*: 98–103.

[10] Kahn MG, Steib SA, Fraser VJ, Dunagan WC. An expert system for culture-based infection-control surveillance. To appear, Symposium on Computer Applications in Medical Care, 1993.

[11] Batini C, Lenzerini M, Navath SB. A comparative analysis of methodologies for database schema integration. ACM Computing Surveys 1986; 323–64.

[12] Kamel MN, Zviran M. Heterogeneous databases integration in a hospital information systems environment: A bottom-up approach. In: Clayton PD, ed. *Proceedings Symposium on Computer Applications in Medical Care*. New York, NY: IEEE Computer Society Press, 1991:363–7.

[13] Gallagher L. Database management standards. Status and applicability. Computer Standards & Interfaces 1991; *12*: 185–92.